Attorney's Docket No.: 07844-235001/P212

#28

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant : M. Hamburg and A.M. Herasimchuk
Art Unit : 2771
Examiner : Ella Colbert
Serial No. : 09/010,801
Filed : January 22, 1998
Title : MAINTAINING DOCUMENT STATE HISTORY

Mail Stop Appeal Brief-Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

BRIEF ON APPEAL

(1) Real Party in Interest

The real party in interest is Adobe Systems Incorporated.

(2) Related Appeals and Interferences

None.

(3) Status of Claims

Claims 1-7, 9-25, 27-37, and 39-47 stand rejected under 35 U.S.C. § 103(a) as unpatentable over U.S. Patent No. 6,018,342 ("Bristor").

Claims 8, 26, and 38 have been cancelled.

(4) Status of Amendments

Amendments to claims 1, 9, 16, 37, 39 – 42, which were made in applicant's response of January 6, 2003, to the Final Office Action mailed on November 4, 2002, have not been entered.

CERTIFICATE OF MAILING BY FIRST CLASS MAIL

I hereby certify under 37 CFR §1.8(a) that this correspondence is being deposited with the United States Postal Service as first class mail with sufficient postage on the date indicated below and is addressed to the Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450.

June 11, 2003
Date of Deposit

Christa Carter
Signature

Christa Carter
Typed or Printed Name of Person Signing Certificate

(5) Summary of Invention

In a first aspect, the invention provides methods and apparatus embodying techniques in which a state history for storing document states is maintained for a document. Whenever an operation by a user that changes the state of the document occurs, the state of the document is automatically captured as it exists after the operation. The captured state is added to the state history. (Specification page 2, lines 11-19; page 7, lines 18-24; page 8, lines 14-23)

In another aspect, the invention provides techniques in which a sequence of commands to change a document is received from a user, and the document state is changed in response to each command. Each time the document state is changed, the changed document state is added to a state history maintained in a computer-readable memory device. For each document state added to the state history, a corresponding entry is added to a history list displayed to the user on a computer-controlled display device operated as part of a graphical user interface. In response to a user action, an item in the history list is selected and the document state corresponding to the selected item in the history list is established as the current state of the document. (Specification page 2, line 26 – page 3, line 9; page 9, lines 2 – 7)

In another aspect, the invention provides techniques including operations to keep a state history of a document, the document states being created automatically whenever a user command to the application changes the state of the document and being complete in themselves. The operations enable the user to discard any of the history to create a revised history. The operation also enables the user to step backward and forward through the history and thereby alter the state of the document to be any of the document states in the revised history, or to designate any one of the document states in the revised history and thereby establish the designated state as the current state of the document. (Specification page 3, lines 23 – 30; page 11, lines 9 - 12)

In another aspect, the invention provides techniques to identify for a user on a display device a set of states that a document has been in by operation of a system. The techniques enable the user to designate any one of the identified states for further operations. For example, the user can establish the designated state as the current state of the document. In another aspect, the techniques include operations to keep a history list for the documents. The user can select a first state from the state history and establish the first selected state of the document as the

current state of the document. The user can select a second state from the state history as a source of data for an operation, the second state being a state created after the first state, and perform the operation with the data from the second state on the first state. (Specification page 3, lines 10 - 16; page 10, line 14 – page 11, line 1)

(6) Issues

1. Are claims 1-7, 9-25, 27-37, and 39-47 properly rejected under 35 U.S.C. § 103(a) as unpatentable over U.S. Patent No. 6,018,342 ("Bristor")?

(7) Grouping of Claims

For the purpose of this appeal, the claims are grouped as follows:

1. For the purpose of Issue 1:
    a. Claims 1-7, 9-25, 27-37, 39-47 stand together.
        i. Claims 9-15, and 39 are separately patentable from claims listed above, at heading "a".
        ii. Claims 16, 22-25, 27-36, 40, 43, and 44 are separately patentable from claims listed above, at heading "a".

(8) Argument

1. Are claims 1-7, 9-25, 27-37, and 39-47 properly rejected under 35 U.S.C. § 103(a) as unpatentable over U.S. Patent No. 6,018,342 ("Bristor")?

Group (a): Claims

Claim 1, as finally rejected, recites a method in which a state history of a document is maintained in a memory, and whenever an operation by a user has occurred that changes the state of the document, the state of the document as it exists after the operation is captured and added to the state history of the document. The balance of the claims in Group (a) incorporate at least the features of maintaining a state history of a document, and may include adding a changed state to the state history.

Bristor does not disclose a state history of a document. With respect to claim 1, the Examiner states, at page 3, that Bristor discloses "maintaining in a memory a state history of a

document" at col. 3, lines 20-35, col. 4, lines 5-27, col. 11, lines 56-67, and col. 12, lines 1-13. The cited portions do not, however, support this assertion. For example, col. 3, lines 20-35 describes a hot key "history mechanism" in which "a user enters one or more characters of a command and . . . [in response] the computer process retrieves and display the most recently entered command which begins with the one or more characters entered by the user." As discussed below, a list of commands executed by a user is not a state history of a document.

Col. 4, lines 5-27 of Bristor describes the "notion of moving backwards or forward within previously retrieved Web documents," while col. 12, lines 1-13, merely describes a computer processor. Similarly, but with respect to claim 16, the Examiner states, at page 7, that Bristor discloses "maintaining in a memory a state history of a document" at col. 3, lines 61-66. The cited portion describes a "list of some of the most recently retrieved Web documents in reverse chronological order." Also similarly, but with respect to claims 17 and 22, the Examiner states, at pages 7 and 9, that Bristor discloses "keeping a history of document state of a document" at col. 4, lines 32-52, and "identifying . . . a set of states that the document has been in" at col. 9, lines 10-31, col. 12, lines 7-8 and lines 31-49. But the cited portions simply describe a list of linked web documents, a system for organizing web documents that have been visited, and a computer system having a mouse. As discussed below, a list of Web documents visited by a user is not a state history of a document.

As described in the applicant's specification, an application that is programmed to implement a state history feature associates with each document a corresponding state history. (Specification page 7, lines 1-3). The application is a computer program designed for a particular task and typically acts on a central depository called a document, whose content a user creates and edits. (Specification page 6, lines 12-15) The state of the document is recorded in the state history after the user makes a change to the document that changes the state of the document. (Specification page 8, lines 19-21) If a user has more than one document open, each has a separate history palette that reflects the state history of the corresponding document. (Specification page 8, lines 2-7) The invention enables a user to access prior states of a document randomly and without regard to intervening states or commands.

Contrary to the Examiner's interpretation, Bristor simply does not disclose maintaining in a memory a state history of a document as described by the applicant. Rather, Bristor provides a

method for organizing commands ("user-generated signals" or "user data") executed by a user on one or more documents. With Bristor's method, a user can re-execute a particular command on the relevant or corresponding document. But the command is executed on the *current* state of the document – which is not necessarily the state of the document that existed when the command was first executed by the user. Bristor does not keep the previous state of the document, and hence Bristor does not disclose a state history as described by the applicant.

Bristor describes, for example and as noted by the Examiner, a history of "URLs" (as in a web browser). By clicking on an URL, a user executes a command ("http://URL") that retrieves the current state of the website specified by the URL. With Bristor's system, a user can save the URL of a visited website in a list, and at some later time, can readily find and select the URL from the list and revisit the website – i.e. retrieve again the document or documents making up the website. But because Bristor's list is a list of commands pertaining to various websites, not the previous state of a website, it cannot be used to retrieve the previous state of the document or documents making up the website. Rather, Bristor's method simply retrieves the website as it currently exists – not necessarily as it was when the user first visited it. If the website has been changed, the new version is retrieved; if it has been deleted, the user may get an error message such as "file not found." Thus, Bristor does not describe or suggest maintaining a state history of a website but, rather, a list of links to the extant states of various websites.

Bristor also describes, for example, a list of "shell" commands (which are executed at the prompt line of certain computer systems). By typing in a shell command, a user can, for example, display the names of the files in a directory or the text of a particular file. With Bristor's system, a user can save all the typed commands in a list and then, at some later time, select one of the commands and re-execute it. But because Bristor's system pertains to various files in a computer system, and not the previous states of a particular document, it cannot be used to retrieve the previous state of a directory or file. Rather, Bristor's system will display the directory or file as it currently exists – not necessarily as it was when the user previously displayed it. Once again, if the contents of the directory or file have been changed or deleted, the user will not be able to retrieve the previous state of the directory or file. Thus, Bristor does not describe or suggest maintaining a state history of a document but, rather, a list of shell commands operable on the current state of various files.

In contrast to the applicant's invention, the lists described by Bristor are histories of a particular user in a computer environment – not histories of the states of a particular document. Accordingly, at least one element of each of the claims in group (a) is not taught or suggested by Bristor, and the applicant respectfully submits that no prima facie case of obviousness under 35 U.S.C. § 103 has been established with respect to these claims.

In addition, it is clear from the language of the claims that the recited methods pertain to computer program applications or computer-implemented methods capable of changing the state of a document and adding the changed state to the state history. The Examiner has admitted in part that Bristor does *not* teach this limitation. The Examiner admits, at page 3 with respect to claim 1, that Bristor does not teach "whenever an interesting operation has occurred, an interesting operation being an operation by a user that changes the state of the document, automatically capturing the state of the document as it exists after the operation and adding the captured state to the state history of the document." Similarly, the Examiner admits, at page 5 with respect to claim 9, that Bristor does not teach "adding [a] changed document state to a state history." The Examiner also admits at page 9 with respect to claim 22 that Bristor does not teach "identifying to the user . . . a set of states that the document has been in and providing the user an editing tool . . ."

The Examiner asserts, at page 4 with respect to claim 9, that Bristor teaches "receiving from the user a sequence of commands to change the document" at col. 6, lines 11-16 and lines 45-61, and "changing the document state in response to each command" at col. 6, lines 29-51. The cited portions, however, merely describe a method for categorizing user commands by associating them with letters of the alphabet – not commands to change the document state. Thus, it is clear that Bristor does not disclose or suggest modifying or changing the state of a document.

Indeed, as discussed above, Bristor's method pertains to the extant state of documents, for example, in a network or computer system, and is completely insensitive to changes in the states of documents. Hence a command that is re-executed from Bristor's users' history list is re-executed on the current state of its target (be it a website, directory, or file) – whether the user desires that or not, and whether the target has been changed or not. Bristor's focus is on providing an efficient way to keep track of user commands so that they can be found and re-

executed using the current state of websites, directories, and files. Bristor is simply not concerned whether changes to the state of a document have occurred, as evidenced by the fact that Bristor does not suggest or describe keeping track of the changes in the state of a document.

The Examiner says that capturing the state of a document after an interesting operation and adding a changed document state to a state history (that is, the elements of claims 1 and 9 that the Examiner admitted were not described by Bristor) would have been obvious to one having ordinary skill in the art at the time the invention was made. The Examiner reasons, at page 3 with respect to claim 1, that "such a modification would allow a user to create a draft ("snapshot") of the current state of the document at a particular point in time", or because, at page 5 with respect to claim 9, "such a modification will allow changes to be made to a document and the changes to be stored . . ." The applicant respectfully disagrees. There is nothing in the Examiner's analysis to support that such a modification to Bristor might be desirable. Indeed, Bristor is not directed to changing the state of a document, let alone keeping track of such changes in any way. The applicant recognizes that the rationale for combining or modifying references can be implied or reasoned from the prior art or from knowledge generally available to one of ordinary skill in the art. But in the absence of any other prior art or a convincing line of reasoning, the only basis for such a modification to Bristor is the hindsight provided by applicant's claims – and the use of hindsight to establish a prima facie case of obviousness is simply not proper. (See *Ex parte Clapp*, 227 USPQ 972, 973 (Bd. Pat. App. & Inter. 1985); see also MPEP 2142, paragraph 2)

Because Bristor fails entirely to disclose or suggest maintaining a state history of a document and also because Bristor fails to disclose or suggest adding a changed state of the document to the state history, at least one element of each of the claims in group (a) is not taught or suggested by Bristor. The applicant therefore respectfully submits that no prima facie case of obviousness under 35 U.S.C. § 103 has been established with respect to these claims. Accordingly, the applicant submits that all of the claims are in condition for allowance.

Sub-Group (a)(i): Claims 9-15 and 39

In addition, claims 9-15 and 39 are separately patentable from the balance of the claims in Group (a). Claim 9 recites a method of interacting with a user who is editing a document, in which a sequence of commands to change the document is received from the user, the document state is changed in response to each command, and the changed document state is added to a state history maintained in a computer-readable memory device each time the document state is changed and, for each document state added to the state history, a corresponding entry is added to a history list displayed to the user on a computer-controlled display device operated as part of a graphical user interface. Then, in response to a user action, the method an item in the history list is selected and the document state corresponding to the selected item in the history list is established as the current state of the document.

Thus, in addition to maintaining a state history and, when the document state is changed, adding the changed state to the state history, claim 9 requires keeping a history list that corresponds to the state history and establishing the document state corresponding to a selected item in the history list as the current state of the document.

The Examiner says that Bristor describes establishing the document state corresponding to the selected item in the history list as the current state of the document at col. 4, lines 42-52. But the cited portion of Bristor describes backtracking through a history of websites visited to view previously visited websites. These different websites are not different document states. Rather, as clearly stated by Bristor in the cited portion, "the history list includes user data specifying hypertext *documents A, B, and F*" such that "the user can follow the history list back to hypertext *document A.*" (emphasis added). Clearly, Bristor describes that the websites are different documents, not different states of a document as described by the applicant.

Thus, while Bristor may describe a history list, it is a history of commands relating to various website documents, which is not the same as a state history of a particular document. And while Bristor does teach selecting an item in the history list and executing the associated command, such an action does not allow the user to return to a former or different state of the document – since the command in Bristor can only operate on the existing state of the document, as discussed previously. Accordingly, applicant submits that claim 9 is allowable over Bristor.

Claims 10-15, and 39 incorporate the features of claim 9, so they are allowable for at least the same reasons set forth above in connection with claim 9.

Group (a)(ii): Claims 16, 22-25, 27-36, 40, 43, and 44

Claims 16, 22-25, 27-36, 40, 43, and 44 stand or fall together. Claim 16 recites a method for editing a document that includes maintaining in a memory a state history of the document. In response to a user action, the method selects a first state from the state history and establishes the first selected state of the document as the current state of the document. The method then, in response to a user action, selects a second state from the state history, the second state being a state created after the first state, as a source of data for an operation, and performs the operation with the data from the second state on the first state. Claim 22 recites a method for controlling an application that creates and modifies a document, including identifying for the user on a display device a set of states that the document has been in by operation of the application, and enabling the user to designate any one of the identified states as a document state operand.

Each of these claims includes maintaining a state history or set of states that a document has been in, and using one of the states as a source of data for an operation on another state or as a document state operand. Bristor does not teach using a second state of a document as a source of data to perform an operation on a different first state of the document. Indeed, because Bristor does not consider changes in the state of a document, nothing in Bristor suggests a second state for a user to designate as a document state operand as required by claim 16, or an editing tool that could take the designated state as a document state operand as required by claim 22. An example of such a tool is the history paintbrush tool described in the Applicant's specification on page 4, lines 28-30, through page 5, lines 1-2.

The Examiner concedes at page 7 that Bristor does not teach (1) performing the operation with the data from the second state on the first state, and concedes at page 9 that Bristor does not teach (2) identifying to the user on a display device a set of state and providing the user an editing tool having the designated state as a document state operand. The Examiner nonetheless asserts that it would have been obvious to one of skill in the art at the time the invention was made to make the first modification "in view of Bristor's teaching of a step backward command" and because such a modification "would allow Bristor's system to allow a user to display and to

retrieve the Web document which immediately follows the currently displayed Web document in the history list." The Examiner then asserts that it would have been obvious to one of skill in the art at the time the invention was made to make the second modification because to do so would allegedly "allow the user to be able to see what commands have been entered and what operations have been performed on the document and the user will be able to use the editing tool to give editing commands provided by the system such as cut, copy, paste, undo, and redo."

The Examiner fails to explain how the use of a step backward command would enable the display of a document following (forward of) a displayed document. The Examiner also fails to explain how providing an editing tool having the designated state as a document state operand would enable the user to cut, copy, or paste. Accordingly, there is nothing in the Examiner's analysis to support that either such modification to Bristor might be desirable. In light of the foregoing, the applicant respectfully submits that no prima facie case of obviousness under 35 U.S.C. § 103 has been established with respect to these claims and, accordingly, claims 16 and 22 are allowable over Bristor.

Claim 40 is a computer program product claim incorporating corresponding features of claim 16 and is allowable for the reasons set forth in connection with claim 16. Claims 23-25, 27-36, 43, and 44 incorporate the features of claim 22 and are allowable for at least the same reasons set forth above in connection with claim 22.

(9) Appendix

Appendix A to this brief is a set of the claims currently pending in this case.

Please apply the brief fee of $320 and a one-month extension fee of $110 and any other charges or credits to Deposit Account No. 06-1050, making reference to attorney docket no. 07844-235001.

Respectfully submitted,

Date: June 11, 2003

Tamara Fraizer
Reg. No. 51,699

Fish & Richardson P.C.
500 Arguello Street, Ste. 500
Redwood City, CA 94063
(650) 839-5070 telephone
(650) 839-5071 facsimile

50154122.2.doc

Appendix of Claims

1. (Previously Amended) A method implemented in a computer program application performing operations on documents having states, the method comprising:

maintaining in a memory a state history of a document; and

whenever an interesting operation has occurred, an interesting operation being an operation by a user that changes the state of the document, automatically capturing the state of the document as it exists after the operation and adding the captured state to the state history of the document.

2. (Original) The method of claim 1, wherein the memory comprises a disk file.

3. (Previously Amended) The method of claim 1, further comprising:

maintaining in the state history the order in which the stored states were automatically added to the state history; and

displaying the state history to a user as a list of document states shown in their stored order.

4. (Previously Amended) The method of claim 3, wherein:

the list of document states displayed to the user comprises a list of items, each item representing a state of the document that existed after an interesting operation and that can be recovered directly by selecting the item.

5. (Previously Amended) The method of claim 4, further comprising:

providing a tool operable under user control to obtain source material from any state in the state history and apply it to a current state of the document, where the document is a raster image.

6. (Previously Amended) The method of claim 4, further comprising:

enabling a user to select any item in the displayed list of items and cause the application to create a new document having the document state corresponding to the selected item.

7. (Original) The method of claim 4, wherein:

each of the captured states in the state history maintains the state data in essentially its original form, whereby the captured state data is suitable for immediate use in other operations.

8. (Cancelled)

9. (Previously Amended) A computer-implemented method of interacting with a user editing a document in a computer program application, the document having a document state, the method comprising:

receiving from the user a sequence of commands to change the document;

changing the document state in response to each command;

adding the changed document state to a state history maintained in a computer-readable memory device each time the document state is changed;

for each document state added to the state history, adding a corresponding entry to a history list displayed to the user on a computer-controlled display device operated as part of a graphical user interface; and

in response to a user action, selecting an item in the history list and establishing the document state corresponding to the selected item in the history list as the current state of the document.

10. (Original) The method of claim 9, wherein:

the state history and the history list are limited to storing a preset number of items and excess items are scrolled off the top of the list as new items are added.

11. (Original) The method of claim 9, wherein:

the state history is stored in a region of memory and the oldest document states in the state history are discarded when free space in the region runs low.

12. (Original) The method of claim 11, wherein:

the oldest document states are found and discarded by a memory management process.

13. (Previously Amended) The method of claim 9, further comprising:

in response to a user command to change the document state corresponding to the selected item in the history list and established as the current state of the document, deleting the items after the selected item in the history list and the corresponding document states from the state history.

14. (Previously Amended) The method of claim 9, further comprising:

in response to a user command to change the document state corresponding to the selected item in the history list and established as the current state of the document, maintaining the items after the selected item in the history list and adding a new item to the end of the history list and a new document state to the state history.

15. (Original) The method of claim 9, further comprising:

enabling a user interface gesture on the history list to create a new document from a document state from the state history.

16. (Previously Amended) A method implemented in a computer program application operable to create and edit a document, comprising:

maintaining in a memory a state history of a document;

in response to a user action, selecting a first state from the state history and establishing the first selected state of the document as the current state of the document;

in response to a user action, selecting a second state from the state history, the second state being a state created after the first state, as a source of data for an operation; and

performing the operation with the data from the second state on the first state.

17. (Previously Amended) A method implemented in a computer program application operable to create and edit a document, comprising:

keeping a history of document states of a document, the document states being created automatically whenever a user command to the application changes the state of the document and being complete in themselves;

enabling the user to discard any of the states in the history to create a revised history; and

enabling the user to step backward and forward through the revised history and thereby alter the state of the document to be any of the document states in the revised history.

18. (Previously Amended) A method implemented in a computer program application operable to create and edit a document, comprising:

keeping a history of document states of a document, the document states being created automatically whenever a user command to the application changes the state of the document and being complete in themselves;

enabling the user to discard any of the states in the history to create a revised history; and

enabling the user to designate any one of the document states in the revised history and thereby establish the designated state as the current state of the document.

19. (Original) The method of claim 18, further comprising:

saving the history when the document is closed on a long-term storage medium, whereby the history may be restored when the document is later opened and across invocations of the application.

20. (Original) The method of claim 19, wherein:

the saved history resides in the document with final document data.

21. (Original) The method of claim 19, wherein:

the saved history resides in a long-term data repository independent of the original document.

22. (Previously Amended) A method enabling a user to control operation of a computer program application for creating and modifying a document, the method comprising:

identifying for the user on a display device a set of states that the document has been in by operation of the application; and

enabling the user to designate any one of the identified states as a document state operand.

23. (Original) The method of claim 22, further comprising:

displaying the document in a user interface window, the document being a digital image.

24. (Original) The method of claim 23, wherein the digital image has a plurality of layers, each of the plurality of layers having a plurality of channels, the method further comprising:

displaying user-interface elements for applying filters to the digital image.

25. (Original) The method of claim 22, further comprising:

establishing the designated state as the current state of the document in response to a user command.

26. (Cancelled)

27. (Amended) The method of claim 22, further comprising:

providing the user a delete tool for deleting the designated state from the set of states.

28. (Original) The method of claim 22, wherein:

the set of states is identified by displaying a scrollable list of elements each identifying one of the states in the set.

29. (Previously Amended) The method of claim 28 wherein the list elements are ordered by the time the corresponding states were created.

30. (Previously Amended) The method of claim 25 wherein the designation and establishment are performed in response to a single command.

31. (Previously Amended) The method of claim 25 wherein the set of states is displayed in an order, the method further comprising:

enabling the user to make a gesture on a user interface indicating a sequence of displayed state identifiers and responding to the gesture by displaying the document in the states indicated as the gesture is made.

32. (Previously Amended) The method of claim 25 further comprising:

enabling the user to modify the document state after the establishing step; and

adding the document state resulting from the modification to the set of states identified on the display device.

33. (Previously Amended) The method of claim 31 wherein the set of states is displayed in order of creation of the states in the set.

34. (Previously Amended) The method of claim 31 wherein the document is a digital image.

35. (Previously Amended) The method of claim 25 further comprising:

providing a step backward and a step forward command for the user to execute to navigate the set of states; and

providing a separate undo and redo command for the user to undo and redo commands entered by the user.

36. (Previously Amended) The method of claim 22, further comprising:

providing a step backward and a step forward command for the user to execute to navigate the set of states; and

providing a separate undo and redo command for the user to undo and redo commands entered by the user.

37. (Previously Amended) Apparatus comprising a computer-readable storage medium tangibly embodying program instructions defining a computer program application for performing operations on documents having states, the program comprising instructions operable for causing a programmable processor to:

maintain in a memory a state history of a document; and

whenever an interesting operation has occurred, an interesting operation being an operation by a user that changes the state of the document, the state being complete in itself and independent of other states; automatically capture the state of the document as it exists after the operation and add the captured state to the state history of the document.

38. (Cancelled)

39. (Previously Amended) Apparatus comprising a computer-readable storage medium tangibly embodying program instructions for interacting with a user editing a document in a computer program application, the document having a document state, the apparatus comprising instructions operable for causing a programmable processor to:

receive from the user a sequence of commands to change the document;

change the document state in response to each command;

add the changed document state to a state history maintained in a computer-readable memory device each time the document state is changed;

for each document state added to the state history, add a corresponding entry to a history list displayed to the user on a computer-controlled display device operated as part of a graphical user interface; and

in response to a user action, select an item in the history list and establish the document state corresponding to the selected item in the history list as the current state of the document.

40. (Previously Amended) A computer program, residing on a computer-readable medium, comprising instructions for causing a computer to:

keep a history list;

in response to a user action, select a first state from the history list and establish the first selected state of the document as the current state of the document;

in response to a user action, select a second state from the history list, the second state being a state created after the first state, as a source of data for an operation; and

perform the operation with the data from the second state on the first state.

41. (Previously Amended) A computer program, residing on a computer-readable medium, comprising instructions for causing a computer to:

keep a history of document states created by a user; the document states being created automatically whenever a user command to the application changes the state of a document and being complete in themselves;

enable the user to discard any of the states in the history to create a revised history; and

enable the user to step backward and forward through the revised history and thereby alter the state of the document to be any of the document states in the revised history.

42. (Previously Amended) A computer program, residing on a computer-readable medium, comprising instructions for causing a computer to:

keep a history of document states created by a user, the document states being created automatically whenever a user command to the application changes the state of a document and being complete in themselves;

enable the user to discard any of the states in the history to create a revised history; and

enable the user to designate any one of the document states in the revised history and thereby establish the designated state as the current state of the document.

43. (Previously Amended) A computer program, residing on a computer-readable medium, comprising instructions for causing a computer to:

create and modify a document;

identify for a user on a display device a set of states that the document has been in by operation of the application; and

enable the user to designate any one of the identified states.

44. (Original) The method of claim 36, further comprising:

providing to the user a first undo command function that operates with reference to the first history and a second undo command function that operates with reference to the second history.

45. (Original) The method of claim 3, further comprising:

establishing as the current state of the document a state stored in the state history.

46. (Original) The method of claim 1, further comprising:

maintaining in memory a history of all operations requested by a user, including operations global to the state of the application.

47. (Original) The apparatus of claim 37, further comprising instructions operable for causing a programmable processor to:

maintain in memory a history of all operations requested by a user, including operations global to the state of the application.